


```

NN      NN      TTTTTTTTTT      000000      AAAAAA      CCCCCCCC      CCCCCCCC      EEEEEEEEEEE      SSSSSSSSS      SSSSSSSSS
NN      NN      TTTTTTTTTT      000000      AAAAAA      CCCCCCCC      CCCCCCCC      EEEEEEEEEEE      SSSSSSSSS      SSSSSSSSS
NN      NN      TT      00      00      AA      AA      CC      CC      EE      SS      SS
NN      NN      TT      00      00      AA      AA      CC      CC      EE      SS      SS
NNNN      NN      TT      00      0000      AA      AA      CC      CC      EE      SS      SS
NNNN      NN      TT      00      0000      AA      AA      CC      CC      EE      SS      SS
NN      NN      TT      00      00      00      AA      AA      CC      CC      EEEEEEEEE      SSSSSSS      SSSSSSS
NN      NN      TT      00      00      00      AA      AA      CC      CC      EEEEEEEEE      SSSSSSS      SSSSSSS
NN      NNNN      TT      0000      00      AAAAAAAAAA      CC      CC      EE      SS      SS
NN      NNNN      TT      0000      00      AAAAAAAAAA      CC      CC      EE      SS      SS
NN      NN      TT      00      00      AA      AA      CC      CC      EE      SS      SS
NN      NN      TT      00      00      AA      AA      CC      CC      EE      SS      SS
NN      NN      TT      000000      AA      AA      CCCCCCCC      CCCCCCCC      EEEEEEEEEEE      SSSSSSSSS      SSSSSSSSS
NN      NN      TT      000000      AA      AA      CCCCCCCC      CCCCCCCC      EEEEEEEEEEE      SSSSSSSSS      SSSSSSSSS

```

....
....
....
....

```

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSS

```

(2)	77	DECLARATIONS
(3)	112	NT\$ASSIGN - NETWORK ASSIGN CHANNEL
(5)	299	NT\$MOD_DEV_CHAR - MODIFY DEVICE CHARACTERISTICS
(6)	394	NT\$MAP_DEV_CHAR - MAP DEVICE CHARACTERISTICS
(7)	484	NT\$RET_DEV_CHAR - RETURN DEVICE CHARACTERISTICS
(8)	569	NT\$ACCESS - PERFORM NETACP ACCESS FUNCTION
(9)	710	NT\$DEACCESS - PERFORM NETACP DEACCESS FUNCTION


```
0000 1          $BEGIN NTOACCESS,000,NF$NETWORK,<NETWORK ACCESS/DEACCESS>
0000 2
0000 3
0000 4 *****
0000 5 *****
0000 6 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 *  ALL RIGHTS RESERVED.
0000 9 *
0000 10 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 *  TRANSFERRED.
0000 16 *
0000 17 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 *  CORPORATION.
0000 20 *
0000 21 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 *
0000 24 *
0000 25 *****
0000 26 *****
0000 27
0000 28 ++
0000 29 Facility: RMS
0000 30
0000 31 Abstract:
0000 32
0000 33 This module performs network access/deaccess functions including
0000 34 assigning a channel to the network device and altering the
0000 35 characteristics of the network device as appropriate. The access
0000 36 function creates a logical link between this process and either:
0000 37 (1) the specified target task, or
0000 38 (2) FAL (file access listener) for remote file access.
0000 39
0000 40 Environment: VAX/VMS, executive mode
0000 41
0000 42 Author: James A. Krycka,      Creation Date: 09-DEC-1977
0000 43
0000 44 Modified By:
0000 45
0000 46 V03-009 JAK0146      J A Krycka      27-Jun-1984
0000 47 Copy entire task specification string to the NCB in NT$ACCESS.
0000 48
0000 49 V03-008 JEJ0029      J E Johnson      17-Apr-1984
0000 50 Fix bug caused by previous change.
0000 51
0000 52 V03-007 JEJ0005      J E Johnson      08-Mar-1984
0000 53 Process quoted filename specifications here to determine if
0000 54 a DAP or NSP request has been made.
0000 55
0000 56 V03-006 JAK0124      J A Krycka      06-Sep-1983
0000 57 Make corresponding source code change for VMS V3.5 patch in
```

0000	58	:	support of VAXELAN.
0000	59	:	
0000	60	:	V03-005 RAS0174 Ron Schaefer 29-Jul-1983
0000	61	:	Change reference of FAB\$V_UFM to FAB\$V_CHAN_MODE.
0000	62	:	
0000	63	:	V03-004 JAK0105 J A Krycka 11-May-1983
0000	64	:	Add comments.
0000	65	:	
0000	66	:	V03-003 KBT0425 Keith B. Thompson 01-Dec-1982
0000	67	:	Change IFB\$W_DEVBUFSIZ to IFB\$L_DEVBUFSIZ.
0000	68	:	
0000	69	:	V03-002 KBT0309 Keith B. Thompson 27-Aug-1982
0000	70	:	Fix some broken branches.
0000	71	:	
0000	72	:	V03-001 JWH0001 Jeffrey W. Horn 29-Jun-1982
0000	73	:	Fix broken BSBW branch.
0000	74	:	
0000	75	:--	

```

0000 77      .SBTTL  DECLARATIONS
0000 78
0000 79 ::
0000 80 :: Include Files:
0000 81 ::
0000 82
0000 83      $DAPPLGDEF      : Define DAP prologue symbols
0000 84      $DAPATTDEF     : Define DAP Attributes message
0000 85      $DEVDEF        : Define Device Characteristics
0000 86      $FABDEF        : Define File Access Block symbols
0000 87      $FWADEF        : Define File Work Area symbols
0000 88      $IFBDEF        : Define IFAB symbols
0000 89      $IODEF         : Define I/O function codes
0000 90      $NAMDEF        : Define NAM block symbols
0000 91      $NWADEF        : Define Network Work Area symbols
0000 92      $PSLDEF        : Define Process Status Longword symbols
0000 93      $RMSDEF        : Define RMS exit code symbols
0000 94
0000 95 ::
0000 96 :: Macros:
0000 97 ::
0000 98      None
0000 99 ::
0000 100 :: Equated Symbols:
0000 101 ::
0000 102
0000 103      ASSUME  DAP$Q_DCODE_FLG EQ 0
0000 104      ASSUME  NWA$Q_FLG EQ 0
0000 105
0000 106 ::
0000 107 :: Own Storage:
0000 108 ::
0000 109      None
0000 110 ::

```



```
0000 112 .SBTTL NT$ASSIGN - NETWORK ASSIGN CHANNEL
0000 113
0000 114 :++
0000 115 : NT$ASSIGN - assigns a channel to the network device (i.e., _NET0:).
0000 116 :
0000 117 : Calling Sequence:
0000 118 :
0000 119 :     BSBW    NT$ASSIGN
0000 120 :
0000 121 : Input Parameters:
0000 122 :
0000 123 :     R8      FAB address
0000 124 :     R9      IFAB address
0000 125 :     R10     FWA address
0000 126 :     R11     Impure Area address
0000 127 :
0000 128 : Implicit Inputs:
0000 129 :
0000 130 :     None
0000 131 :
0000 132 : Output Parameters:
0000 133 :
0000 134 :     R0      Status code (SS)
0000 135 :     R1-R3   Destroyed
0000 136 :
0000 137 : Implicit Outputs:
0000 138 :
0000 139 :     FAB$B_ACMODES
0000 140 :     FWASV_OBJTYPE
0000 141 :     IFB$W_CHNL
0000 142 :     IFB$B_MODE
0000 143 :     IFB$L_NWA_PTR
0000 144 :     NWA$Q_NCB
0000 145 :
0000 146 : Completion Codes:
0000 147 :
0000 148 :     Standard system service status codes
0000 149 :
0000 150 : Side Effects:
0000 151 :
0000 152 :     None
0000 153 :
0000 154 : --
0000 155 :
0000 156 NT$ASSIGN::
0000 157     BBC      #FWASV_QUOTED,(R10),5$ ; Entry point
0004 158     BSBB     PARSE_QUOTED_STRING ; Not quoted, must be DAP request
0006 159     BLBC     R0,20$ ; Parse the quoted specification
0009 160 ; If parse failed, return the error
0009 161 :
0009 162 : Setup device name descriptor and associated string for input to the
0009 163 : Assign Channel system service.
0009 164 :
0009 165 :
0009 166 5$:    MOVL    IFB$L_NWA_PTR(R9),R1 ; Get address of NWA
000D 167     MOVAQ    NWA$Q_NCBTR1),R2 ; Get address of scratch descriptor
0012 168     MOVAB     NWA$T_NCBBUF(R1),R3 ; Get address of scratch buffer
```

05 6A 1A E1 0000 157
48 50 4C 10 0004 158
48 50 E9 0006 159

51 3C A9 D0 0009 166
52 0264 C1 7E 000D 167
53 052C C1 9E 0012 168

```
63 20203A30 54454E5F 8F 62 06 D0 0017 169      MOVL #6,(R2)      ; Fill in string count
                                04 A2 53 D0 001A 170      MOVL R3,4(R2)     ; Fill in string address
                                7D 001E 171      MOVQ #^A\_NET0: \,(R3) ; Store device name string
                                0029 172
                                0029 173
                                0029 174      ; Determine which mode to use in making the channel assignment.
                                0029 175
                                0029 176
                                51 01 D0 0029 177      MOVL #PSL$C_EXEC,R1      ; Assume executive mode
                                11 E1 002C 178      BBC #FAB$V_UFO,-      ; Branch if UFO clear
                                10 04 A8 002E 179      FAB$F_FOP(R8),10$
                                02 EF 0031 180      EXTZV #FAB$V_CHAN_MODE,-      ; Get requested mode
                                02 0033 181      #FAB$S_CHAN_MODE,-
                                51 4A A8 0034 182      FAB$B_ACMODES(R8),R1
                                51 0A A9 91 0037 183      CMPB IFB$B_MODE(R9),R1      ; Maximize this with caller's mode
                                04 1B 003B 184      BLEQU 10$
                                51 0A A9 9A 003D 185      MOVZBL IFB$B_MODE(R9),R1      ; Switch to caller's mode
                                0041 186
                                0041 187
                                0041 188      ; Create a control/information path to NETACP in preparation for
                                0041 189      ; non-transparent network I/O. Do not associate a mailbox with the channel,
                                0041 190      ; as secondary inbound connect initiates and interrupt messages are not
                                0041 191      ; supported.
                                0041 192
                                0041 193
                                0041 194 10$: $ASSIGN_S-      ; Assign the channel
                                0041 195      DEVNAM=(R2)-      ; Address of device name descriptor
                                0041 196      CHAN=IFB$V_CHNL(R9)-      ; Address to return channel #
                                0041 197      ACMODE=R1      ; Access mode
                                05 0051 198 20$: RSB      ; Exit with system service code in R0
```



```
0052 200 :++
0052 201 : Parse the quoted file name string to determine the network specific
0052 202 : information.
0052 203 :
0052 204 : (1) Set the OBJTYPE flag if the quoted string contains an equals character
0052 205 : to indicate that it is of the "objecttype=..." form. This is form 1.
0052 206 :
0052 207 : (2) Set the NETSTR flag if the quoted string contains a slash character
0052 208 : after the equals character to indicate that it is of the
0052 209 : "objecttype=taskname/netacp_data" form. Note that the logical name
0052 210 : translation of SYSSNET yields an equivalence string containing a
0052 211 : quoted string of this form. This is form 2.
0052 212 :
0052 213 : (3) In addition, if the quoted string is of the form given in 3, then
0052 214 : store a character count that represents the number of characters
0052 215 : in the substring /netacp_data" (including the trailing quote).
0052 216 :
0052 217 : (4) Set the WILDCARD character flag if test 1 failed and the quoted string
0052 218 : contains one of the following characters: asterisk, percent sign, or
0052 219 : question mark.
0052 220 :
0052 221 : (5) Finally, copy the quoted string (including the quotes) to buffer in NWA.
0052 222 :
0052 223 :--
0052 224 :
0052 225 PARSE_QUOTED_STRING: ; Entry point
0052 226 PUSHF #^M<R4,R5,R6,R7> ; Save working registers
56 00F0 8F BB 0056 227 MOVQ FWASQ_QUOTED(R10),R6 ; Get the quoted string descriptor.
005B 228 :
005B 229 :
005B 230 : Check for a task specification string enclosed in quotes, i.e.,
005B 231 : node::"objecttype=..." form. (1) or (2).
005B 232 :
005B 233 :
005B 234 LOCC #^A\=\,R6,(R7) ; Search for '=' within quoted string
67 56 3D 3A 005B 235 BEQL 10$ ; Branch if no match (R0=0 on no match)
005F 236 SSB #FWASV_OBJTYPE,(R10) ; Flag 'objecttype=...' form
0061 237 : of quoted string
0065 238 :
0065 239 :
0065 240 : Now check if it is of form (2).
0065 241 :
0065 242 :
0065 243 LOCC #^A\/\,R0,(R1) ; Search for '/' within quoted string
61 50 2F 3A 0065 244 BEQL 30$ ; Branch if no match
0069 245 CMPW R0,#4 ; Length of /netacp_data" must be
006B 246 BLSSU ERRQUO ; at least 4 characters
006E 247 SSB #FWASV_NETSTR,(R10) ; Flag 'objecttype=taskname/...' form
0070 248 BRB 30$ ; of quoted string
0074 249 :
0076 250 :
0076 251 : Check for a wildcard foreign file specification string enclosed in quotes,
0076 252 : i.e., node::"foreign-filespec" form.
0076 253 :
0076 254 : Note: Since the parse operation is perform entirely at the local node, there
0076 255 : is no way to tell for sure whether or not the file specification
0076 256 : contains any wildcard characters (as understood by the remote system).
```

```
0076 257 ; The best we can do here is to scan the quoted string for likely wildcard
0076 258 ; characters (*, %, and ?), and set the FWASV_WILDCARD bit if any are
0076 259 ; found.
0076 260 ;
0076 261 ;
67 56 2A 3A 0076 262 10$: LOCC #^A\*\,R6,(R7) ; Search for '*' within quoted string
0C 12 007A 263 BNEQ 20$ ; and branch if found
67 56 25 3A 007C 264 LOCC #^A\%\,R6,(R7) ; Search for '%' within quoted string
06 12 0080 265 BNEQ 20$ ; and branch if found
67 56 3F 3A 0082 266 LOCC #^A\?\,R6,(R7) ; Search for '?' within quoted string
16 13 0086 267 BEQL 30$ ; and branch if not found
0088 268 ;
0088 269 ;
0088 270 ; Store the wildcard status here as a generic wildcard only.
0088 271 ;
0088 272 ;
57 28 A8 D0 0088 273 20$: SSB #FWASV_WILDCARD,(R10) ; Signal wildcard character present
00000000'EF 16 008C 274 MOVL FAB$L_NAM(R8),R7 ; Get the name block pointer
05 50 E9 0090 275 JSB RMS$CHKNAM ; Ensure that the NAM block is good.
0096 276 BLBC R0,30$ ; Branch if not.
0099 277 SSB #NAMS$V_WILDCARD,- ; Declare the generic wildcard.
0099 278 NAMS$L_FNB(R7) ;
009E 279 ;
009E 280 ;
009E 281 ; Save the number of bytes in the optional "/netacp_data" in the NWA.
009E 282 ;
009E 283 ;
52 3C A9 D0 009E 284 30$: MOVL IFB$L_NWA_PTR(R9),R2 ; get address of NWA
016F C2 50 90 00A2 285 MOV B R0,NWASB_NETSTRSIZ(R2) ; store # bytes in /netacp_data"
00A7 286 ;
00A7 287 RMSSUC ;
00F0 8F BA 00AA 288 POPR #^M<R4,R5,R6,R7> ; Restore working registers
05 00AE 289 RSB ; Exit successfully
00AF 290 ;
00AF 291 ;
00AF 292 ; Process quoted string error.
00AF 293 ;
00AF 294 ;
00F0 8F BA 00AF 295 ERRQUO: RMSERR QUO ; Declare error in quoted string
05 00B4 296 POPR #^M<R4,R5,R6,R7> ; Restore working registers
00B8 297 RSB
```

```
00B9 299 .SBTTL NT$MOD_DEV_CHAR - MODIFY DEVICE CHARACTERISTICS
00B9 300
00B9 301 :++
00B9 302 : NT$MOD_DEV_CHAR - modifies the characteristics of the network device and
00B9 303 : denotes whether network I/O will be performed at the DAP or NSP level.
00B9 304 :
00B9 305 : Calling Sequence:
00B9 306 :
00B9 307 :     BSBW    NT$MOD_DEV_CHAR
00B9 308 :
00B9 309 : Input Parameters:
00B9 310 :
00B9 311 :     R8      FAB address
00B9 312 :     R9      IFAB address
00B9 313 :     R10     FWA address
00B9 314 :     R11     Impure Area address
00B9 315 :
00B9 316 : Implicit Inputs:
00B9 317 :
00B9 318 :     FWASV_OBJTYPE
00B9 319 :     FWASB_SUBNODCNT
00B9 320 :
00B9 321 : Output Parameters:
00B9 322 :
00B9 323 :     None
00B9 324 :
00B9 325 : Implicit Outputs:
00B9 326 :
00B9 327 :     IFBSV_DAP
00B9 328 :     IFBSV_NSP
00B9 329 :     IFBSL_AS_DEV
00B9 330 :     IFBSL_PRIM_DEV
00B9 331 :     IFBSL_DEVBDFSIZ
00B9 332 :
00B9 333 : Completion Codes:
00B9 334 :
00B9 335 :     None
00B9 336 :
00B9 337 : Side Effects:
00B9 338 :
00B9 339 :     None
00B9 340 :
00B9 341 : --
00B9 342 :
00B9 343 : NT$MOD_DEV_CHAR::                                ; Entry point
00B9 344 :
00B9 345 :
00B9 346 : Determine whether the network request is for file access via a remote FAL or
00B9 347 : for task-to-task communication.
00B9 348 :
00B9 349 : Note: If more than one node spec string was specified (manual routing), then
00B9 350 : treat this as a DAP level access so that RMS will connect to FAL at the
00B9 351 : adjacent node (which is actually an intermediate node).
00B9 352 :
00B9 353 :
00B9 354 :     TSTB    FWASB_SUBNODCNT(R10)                ; Branch if more than one node spec
00B9 355 :     BNEQ    10$                                  ; was specified
```

2F AA 95
04 12 00BC


```
OC 6A 31 E0 00BE 356 BBS #FWASV_OBJTYPE,(R10),20$; Branch if device name string is
00C2 357 ; in NSP objtype type format
00C2 358
00C2 359
00C2 360
00C2 361 Denote that network I/O thru RMS will be at the file access level using
00C2 362 DAP to communicate with the remote FAL.
00C2 363
00C2 364 Note: IFB$L_DEVBUSIZ may be adjusted upward by NT$EXCH_CNF after the
00C2 365 negotiated DAP buffer size has been determined. For record I/O mode
00C2 366 access to a remote file, IFB$L_DEVBUSIZ limits the maximum record size
00C2 367 that can be supported because it is used at $CONNECT time to allocate
00C2 368 the BDB buffer through which all records must pass.
00C2 369
00C2 370 10$: $SETBIT #IFB$V_DAP,(R9) ; Set DAP flag
48 A9 0200 8F 3C 00C6 371 MOVZWL #512,IFB$L_DEVBUSIZ(R9); Establish initial device buffer size
0A 11 00CC 372 BRB 30$ ; Join common code
00CE 373
00CE 374
00CE 375 Denote that network I/O thru RMS will be at the task-to-task level using
00CE 376 NSP to communicate with the remote partner process.
00CE 377
00CE 378
00CE 379 20$: $SETBIT #IFB$V_NSP,(R9) ; Set NSP flag
48 A9 1000 8F 3C 00D2 380 MOVZWL #4096,IFB$L_DEVBUSIZ(R9); Establish device buffer size
00D8 381
00D8 382
00D8 383 Alter the device characteristics.
00D8 384
00D8 385
00D8 386 30$: $SETBIT #DEV$V_REC,IFB$L_PRIM_DEV(R9)
00DC 387 ; Say its a record oriented device
00DC 388 $CLRBIT #DEV$V_MBX,IFB$L_PRIM_DEV(R9)
00E0 389 ; Say its a not a mailbox-like device
008C 69 D0 00E0 390 MOVL IFB$L_PRIM_DEV(R9),- ; Copy device characteristics
C9 00E2 391 IFB$L_AS_DEV(R9)
05 00E5 392 40$: RSB ; Exit
```

```
00E6 394 .SBTTL NT$MAP_DEV_CHAR - MAP DEVICE CHARACTERISTICS
00E6 395
00E6 396 :++
00E6 397 : NT$MAP_DEV_CHAR - takes the device characteristics returned by FAL and maps
00E6 398 : them into an RMS bit pattern, then saves them in NWA for use later.
00E6 399 : If the remote device is a terminal or mailbox, this is noted also.
00E6 400
00E6 401 : Calling Sequence:
00E6 402
00E6 403 : BSBW NT$MAP_DEV_CHAR
00E6 404
00E6 405 : Input Parameters:
00E6 406
00E6 407 : R7 NWA (=DAP) address
00E6 408 : R8 FAB address
00E6 409 : R9 IFAB address
00E6 410 : R10 FWA address
00E6 411 : R11 Impure Area address
00E6 412
00E6 413 : Implicit Inputs:
00E6 414
00E6 415 : DAP$L_DEV
00E6 416 : DAP$V_DEV
00E6 417
00E6 418 : Output Parameters:
00E6 419
00E6 420 : R0-R2 Destroyed
00E6 421
00E6 422 : Implicit Outputs:
00E6 423
00E6 424 : NWA$L_DEV
00E6 425 : NWA$V_DEVCHAR
00E6 426 : NWA$V_DEVMBX
00E6 427 : NWA$V_DEVTRM
00E6 428
00E6 429 : Completion Codes:
00E6 430
00E6 431 : None
00E6 432
00E6 433 : Side Effects:
00E6 434
00E6 435 : None
00E6 436
00E6 437 :--
00E6 438
00E6 439 NT$MAP_DEV_CHAR::
00E6 440 BBS #DAP$V_DEV,- : Entry point
00E8 441 DAP$L_ATTMENU(R7),10$ : Branch if partner returned device
00EB 442 RSB : characteristics
00EC 443 10$: $SETBIT #NWA$V_DEVCHAR,(R7) : Exit
00F0 444 : Flag receipt of characteristics
00F0 445
00F0 446 : Map DAP bit definitions into RMS bit definitions for the field and store them
00F0 447 : in NWA for use later.
00F0 448
00F0 449
51 68 A7 D0 00F0 450 MOVL DAP$L_DEV(R7),R1 : Get DEV bits returned by FAL
```

```
52  D4 00F4 451  CLRL R2 : Clear corresponding RMS bits
      00F6 452  $MAPBIT DAPSV_DEVREC,DEVSV_REC : Map REC bit
      00FE 453  $MAPBIT DAPSV_DEVCCL,DEVSV_CCL : Map CCL bit
      0106 454  $MAPBIT DAPSV_DEVTRM,DEVSV_TRM : Map TRM bit
      010E 455  $MAPBIT DAPSV_DEVDIR,DEVSV_DIR : Map DIR bit
      0116 456  $MAPBIT DAPSV_DEVSDI,DEVSV_SDI : Map SDI bit
      011E 457  $MAPBIT DAPSV_DEVSQD,DEVSV_SQD : Map SQD bit
      0126 458  $MAPBIT DAPSV_DEVSPL,DEVSV_SPL : Map SPL bit
      012E 459  $MAPBIT DAPSV_DEVNET,DEVSV_NET : Map NET bit
      0136 460  $MAPBIT DAPSV_DEVFOD,DEVSV_FOD : Map FOD bit
      013E 461  $MAPBIT DAPSV_DEVSHR,DEVSV_SHR : Map SHR bit
      0146 462  $MAPBIT DAPSV_DEVGEN,DEVSV_GEN : Map GEN bit
      014E 463  $MAPBIT DAPSV_DEVAVL,DEVSV_AVL : Map AVL bit
      0156 464  $MAPBIT DAPSV_DEVMNT,DEVSV_MNT : Map MNT bit
      015E 465  $MAPBIT DAPSV_DEVMBX,DEVSV_MBX : Map MBX bit
      0166 466  $MAPBIT DAPSV_DEVDMT,DEVSV_DMT : Map DMT bit
      016E 467  $MAPBIT DAPSV_DEVELG,DEVSV_ELG : Map ELG bit
      0176 468  $MAPBIT DAPSV_DEVALL,DEVSV_ALL : Map ALL bit
      017E 469  $MAPBIT DAPSV_DEVFOR,DEVSV_FOR : Map FOR bit
      0186 470  $MAPBIT DAPSV_DEVSWL,DEVSV_SWL : Map SWL bit
      018E 471  $MAPBIT DAPSV_DEVIDV,DEVSV_IDV : Map IDV bit
      0196 472  $MAPBIT DAPSV_DEVODV,DEVSV_ODV : Map ODV bit
      019E 473  $MAPBIT DAPSV_DEVRND,DEVSV_RND : Map RND bit
      01A6 474  $MAPBIT DAPSV_DEVRTM,DEVSV_RTM : Map RTM bit
      01AE 475  $MAPBIT DAPSV_DEVRCK,DEVSV_RCK : Map RCK bit
      01B6 476  $MAPBIT DAPSV_DEWCK,DEVSV_WCK : Map WCK bit
00C0 C7 52  D0 01BE 477  MOVL R2,NW$SL_DEV(R7) : Save characteristics for use later
      04 52 02  E1 01C3 478  BBC #DEVSV_TRM,R2,20$ : Branch if device is not a terminal
      01C7 479  $SETBIT #NW$SV_DEVTRM,(R7) : Flag remote device as a terminal
      04 52 14  E1 01CB 480 20$: BBC #DEVSV_MBX,R2,30$ : Branch if device is not a mailbox
      01CF 481  $SETBIT #NW$SV_DEVMBX,(R7) : Flag remote device as a mailbox
      05 01D3 482 30$: RSB : Exit
```



```

01D4 484 .SBTTL NTSRET_DEV_CHAR - RETURN DEVICE CHARACTERISTICS
01D4 485
01D4 486 :++
01D4 487 NTSRET_DEV_CHAR - returns the true device characteristic information to the
01D4 488 user's FAB iff all of the following conditions are met:
01D4 489 (1) FAL returned device characteristics in the DAP Attributes message.
01D4 490 (2) FAL is implemented to DAP V5.6 or later.
01D4 491 (3) The remote node is running VAX/VMS or VAXELAN or the file accessed
01D4 492 is a relative or an indexed file.
01D4 493
01D4 494 This restriction is here to accomodate the VMS COPY utility which uses
01D4 495 the FAB$V_BRO and RAB$V_BIO options to defer the decision of whether
01D4 496 to use record I/O ($GET7$PUT) or block I/O ($READ/$WRITE) for sequential
01D4 497 files until $CONNECT time. COPY always uses block I/O to transfer
01D4 498 relative and indexed files.
01D4 499
01D4 500 COPY examines the device characteristics returned on $OPEN (and $CREATE)
01D4 501 in conjunction with other inputs to determine whether to use record
01D4 502 or block I/O. Since it does not know at open time if block I/O will be
01D4 503 used, it sets the FAB$V_BRO bit in FAB$B_FAC on $OPEN, and if block I/O
01D4 504 mode is chosen, it sets the RAB$V_BIO bit in RAB$L_ROP on $CONNECT to
01D4 505 specify block I/O mode.
01D4 506
01D4 507 Note that NT$GET_FAC_SHR and NT$ENCODE_ROP send the DAP$V_BRO and
01D4 508 DAP$V_ROPBIO bits, respectively, based on a similar system specific
01D4 509 check.
01D4 510
01D4 511 Note that the algorithm used to return device characteristics may become
01D4 512 less restrictive in the future, especially if COPY is modified to avoid
01D4 513 using the FAB$V_BRO and RAB$V_BIO options.
01D4 514
01D4 515 Calling Sequence:
01D4 516 BSBW NTSRET_DEV_CHAR
01D4 517
01D4 518 Input Parameters:
01D4 519
01D4 520 R8 FAB address
01D4 521 R9 IFAB address
01D4 522 R10 FWA address
01D4 523 R11 Impure Area address
01D4 524
01D4 525 Implicit Inputs:
01D4 526
01D4 527 DAP$V_GEQ_V56
01D4 528 DAP$V_VAXVMS, DAP$V_VAXELAN
01D4 529 IFB$L_NWA_PTR
01D4 530 NWA$B_ORG
01D4 531 NWA$L_DEV
01D4 532 NWA$V_DEVCHAR
01D4 533
01D4 534 Output Parameters:
01D4 535
01D4 536 R1 Destroyed
01D4 537
01D4 538 Implicit Outputs:
01D4 539
01D4 540

```

```

01D4 541 : FAB$L_DEV
01D4 542 : FAB$L_SDC
01D4 543 :
01D4 544 : Completion Codes:
01D4 545 :
01D4 546 : None
01D4 547 :
01D4 548 : Side Effects:
01D4 549 :
01D4 550 : None
01D4 551 :
01D4 552 : --
01D4 553 :
01D4 554 NTSRET_DEV_CHAR::
51 3C A9 D0 01D4 555 MOVL IFB$L_NWA_PTR(R9),R1 : Entry point
24 61 11 E1 01D8 556 BBC #NWA$V_DEVCHAR,(R1),20$ : Get address of NWA (and DAP)
01DC 557 : Branch if partner did not return
20 61 24 E1 01DC 558 BBC #DAP$V_GEQ_V56,(R1),20$ : device characteristic information
0B 61 34 E0 01E0 559 BBS #DAP$V_VAXVMS,(R1),10$ : Branch if partner uses DAP before V5.6
07 61 35 E0 01E4 560 BBS #DAP$V_VAXELAN,(R1),10$ : Branch if partner is VAX/VMS
00 00C6 C1 91 01E8 561 CMPB NWA$B_ORG(R1),#NWA$B_SEQ : Branch if partner is VAXELAN
51 00C0 C1 D0 01ED 562 BEQL 20$ : Branch if SEQ organization
40 A8 51 D0 01EF 563 10$: MOVL NWA$L_DEV(R1),R1 : else fall thru if REL or IDX
44 A8 51 D0 01F4 564 $SETBIT #DEV$V_NET,R1 : Get actual device characteristics
05 0200 565 MOVL R1,FAB$L_DEV(R8) : Declare this a remote network device
566 MOVL R1,FAB$L_SDC(R8) : Update user DEV field in FAB
567 20$: RSB : Update user SDC field in FAB
: Exit

```

```

0201 569 .SBTTL NT$ACCESS - PERFORM NETACP ACCESS FUNCTION
0201 570
0201 571 :++
0201 572 : NT$ACCESS - creates a logical link.
0201 573
0201 574 : Calling Sequence:
0201 575
0201 576 : BSBW NT$ACCESS
0201 577
0201 578 : Input Parameters:
0201 579
0201 580 : R8 FAB address
0201 581 : R9 IFAB address
0201 582 : R10 FWA address
0201 583 : R11 Impure Area address
0201 584
0201 585 : Implicit Inputs:
0201 586
0201 587 : IFBSV_DAP
0201 588 : IFBSW_CHNL
0201 589 : FWASQ_NODE
0201 590 : FWASV_NETSTR
0201 591 : NWSAQ_QUOTED
0201 592
0201 593 : Output Parameters:
0201 594
0201 595 : R0 Status code (SS)
0201 596 : R1-R6 Destroyed
0201 597 : AP Destroyed
0201 598
0201 599 : Implicit Outputs:
0201 600
0201 601 : IFBSL_IOS
0201 602 : IFBSV_ACCESSED
0201 603 : NWSAQ_NCB
0201 604
0201 605 : Completion Codes:
0201 606
0201 607 : Standard system service status codes
0201 608
0201 609 : Side Effects:
0201 610
0201 611 : None
0201 612
0201 613 :--
0201 614
0201 615 NT$ACCESS:: ; Entry point
0201 616
0201 617 :+
0201 618 : Build a Network Connect Block (NCB) to be used as input for the NETACP
0201 619 : access function.
0201 620
0201 621 : The NCB consists of a string with the following general syntax:
0201 622
0201 623 : nodename'access_control_string'::'objecttype=taskid/netacp_data'
0201 624
0201 625 : Where:

```



```
0201 626 : (1) the access control string in the node spec is present only if provided
0201 627 : by the user (directly or via logical name translation).
0201 628 : (2) for accessing a remote file, the quoted string used by RMS is "FAL="
0201 629 : (to request the services of the remote file access listener).
0201 630 : (3) for user task-to-task communication, the quoted string used is the one
0201 631 : supplied which may include an optional data counted string. Note that
0201 632 : the logical name SYS$NET used as a file specification translates to a
0201 633 : string of the node::"objecttype=..." form.
0201 634 :
0201 635 : Obtain the node spec string.
0201 636 :
0201 637 :
51 3C A9 D0 0201 638 : MOVL IFBSL_NWA_PTR(R9),R1 : Get address of NWA
56 0264 C1 7E 0205 639 : MOVAB NWA$Q_NCB(R1),R6 : Get address of scratch descriptor
53 052C C1 9E 020A 640 : MOVAB NWA$Q_NCBBUF(R1),R3 : Get address of scratch buffer
04 A6 53 D0 020F 641 : MOVL R3,4(R6) : Fill in descriptor address
01B4 CA 28 0213 642 : MOVCL FWA$Q_NODE1(R10),- : Copy primary (first) node spec string
63 01B8 DA 0217 643 : @FWA$Q_NODE1+4(R10),(R3); to NCB including double colon
021B 644 :
021B 645 : Obtain the quoted string.
021B 646 :
021B 647 : Determine whether the network request is for file access via a remote FAL or
021B 648 : for task-to-task communication.
021B 649 :
021B 650 :
10 69 3E E0 021B 651 : BBS #IFBSV_DAP,(R9),10$ : Branch if file access request
021F 652 :
021F 653 : It is a task-to-task communication request.
021F 654 :
021F 655 :
021F 656 :
021F 657 :
021F 658 : $STSTPT NTACC_NSP :
0225 659 : MOVCL FWA$Q_QUOTED(R10),- : Append quoted string to NCB
63 0170 CA 28 0229 660 : @FWA$Q_QUOTED+4(R10),(R3)
0174 DA 11 022D 661 : BRB 20$ : We're finished building NCB
12 022F 662 :
022F 663 : It is a remote file access request.
022F 664 :
022F 665 :
022F 666 :
022F 667 10$: $STSTPT NTACC_DAP :
83 4C414622 8F D0 0235 668 : MOVL #^A\FAL\,(R3)+ : Request object type FAL
83 223D 8F B0 023C 669 : MOVW #^A\='',(R3)+ :
0241 670 :
0241 671 : Calculate the size of the NCB and store it in the NCB descriptor block.
0241 672 :
0241 673 :
0241 674 :
66 53 04 A6 C3 0241 675 20$: SUBL3 4(R6),R3,(R6) : Fill in descriptor size
0246 676 :
0246 677 :
0246 678 : Perform the NETACP access function. It will be either an NSP connect
0246 679 : initiate or an NSP connect confirm function. Both use the same function and
0246 680 : subfunction codes, but NETACP differentiates based on the context of the call
0246 681 : (whether or not the taskspec string in the NCB contains a slash followed by
0246 682 : a two-byte nonzero DECnet link identifier).
```

```
00000000'EF 16 0246 683 :-  
52 8E D0 0246 684  
51 56 D0 024C 685 JSB RMS$SETEFN ; Request event flag number to use  
56 8ED0 024F 686 MOVL (SP)+,R2 ; and store it  
0252 687 MOVL R6,R1 ; Copy address of NCB descriptor  
0255 688 POPL R6 ; Save return PC  
0255 689 $QIO_S- ; Issue connect initiate/confirm  
0255 690 EFN=R2- ; Event flag #  
0255 691 CHAN=IFB$W_CHNL(R9)- ; Channel #  
0255 692 FUNC=#IOS_ACCESS!IOSM_ACCESS- ; Function code  
0255 693 IOSB=IFB$C_IOS(R9)- ; I/O status block  
0255 694 ASTADR=L^RMS$STALLAST- ; AST address  
0255 695 ASTPRM=R9- ; AST parameter  
0255 696 P1=0- ; Must be zero  
0255 697 P2=R1 ; Address of NCB descriptor  
00000000'EF 16 027A 698 BLBC R0,30$ ; Branch on failure  
04 50 E9 027D 699 JSB RMS$STALL ; Await completion  
04 50 E9 0283 700 BLBC R0,30$ ; Branch on failure  
0286 701  
0286 702  
0286 703 : State that ACP access function has been performed successfully. This will  
0286 704 : trigger the network deaccess code at close time.  
0286 705 :  
0286 706  
0286 707 $SETBIT #IFB$V_ACCESSED,(R9) ; File has been accessed  
66 17 028A 708 30$: JMP (R6) ; Return to caller
```

```

028C 710 .SBTTL NT$DEACCESS - PERFORM NETACP DEACCESS FUNCTION
028C 711
028C 712 :++
028C 713 : NT$DEACCESS - destroys a logical link.
028C 714 :
028C 715 : Calling Sequence:
028C 716 :
028C 717 :     BSBW    NT$DEACCESS
028C 718 :
028C 719 : Input Parameters:
028C 720 :
028C 721 :     R8      FAB address
028C 722 :     R9      IFAB address
028C 723 :     R10     FWA address
028C 724 :     R11     Impure Area address
028C 725 :
028C 726 : Implicit Inputs:
028C 727 :
028C 728 :     IFBSW_CHNL
028C 729 :
028C 730 : Output Parameters:
028C 731 :
028C 732 :     R0      Status code (SS)
028C 733 :     R1-R4   Destroyed
028C 734 :     AP      Destroyed
028C 735 :
028C 736 : Implicit Outputs:
028C 737 :
028C 738 :     IFBSL_IOS
028C 739 :
028C 740 : Completion Codes:
028C 741 :
028C 742 :     Standard system service status codes
028C 743 :
028C 744 : Side Effects:
028C 745 :
028C 746 :     None
028C 747 :
028C 748 : --
028C 749 :
028C 750 NT$DEACCESS::
028C 751     STSTPT  NTDEACCES      : Entry point
028C 752     POPL   R4             : Save return PC
028C 753     JSB   RM$SETEFN      : Request event flag number to use
028C 754     MOVL (SP)+,R2      : and store it
028C 755     $QIO_S-             : Issue synchronous disconnect
028C 756     EFN=R2-             : Event flag #
028C 757     CHAN=IFBSW_CHNL(R9)- : Channel #
028C 758     FUNC=#IOS_DEACCESS!IOSM- : Function code
028C 759     IOSB=IFBSL_IOS(R9)-  : I/O status block
028C 760     ASTADR=L*RM$STALLAST- : AST address
028C 761     ASTPRM=R9-           : AST parameter
028C 762     P1=0-               : Must be zero
028C 763     P2=#0              : Specify no userdata to return
028C 764     BLBC  R0,IOS       : Branch on failure
028C 765     JSB   RM$STALL     : Await completion
028C 766     JMP  (R4)        : Return to caller

```

00000000 54 BED0 16 0292 752
 52 : 8E D0 0295 753
 029B 754
 029E 755
 029E 756
 029E 757
 029E 758
 029E 759
 029E 760
 029E 761
 029E 762
 029E 763
 06 50 E9 02C3 764
 00000000 54 BED0 16 02C6 765
 64 17 02CC 766 10\$: JMP (R4)

NTOACCESS
V04-000

J 6
NETWORK ACCESS/DEACCESS

NTSDEACCESS - PERFORM NETACP DEACCESS FU

15-SEP-1984 23:47:02

VAX/VMS Macro V04-00

Page 18
(9)

5-SEP-1984 16:20:08

[RMS.SRC] NTOACCESS.MAR;1

02CE 767
02CE 768

.END

; End of module

NTOACCESS
Symbol table

NETWORK ACCESS/DEACCESS

K 6

15-SEP-1984 23:47:02 VAX/VMS Macro V04-00
5-SEP-1984 16:20:08 [RMS.SRC]NTOACCESS.MAR;1

Page 19
(9)

```

$$PSECT_EP      = 00000000
$$RMSTEST       = 0000001A
$$RMS_PBUGCHK   = 00000010
$$RMS_TBUGCHK   = 00000008
$$RMS_UMODE     = 00000004
$$T1            = 00000000
DAP$B_BKS       = 00000050
DAP$B_BSZ       = 00000052
DAP$B_DATATYPE  = 00000044
DAP$B_DCODE_FID = 00000019
DAP$B_DCODE_MAC = 0000001B
DAP$B_DCODE_MSG = 0000001A
DAP$B_FSZ       = 00000051
DAP$B_ORG       = 00000045
DAP$B_RAT       = 00000047
DAP$B_RFM       = 00000046
DAP$B_X_FIELD   = 00000024
DAP$C_BCN       = 000000C0
DAP$K_BLN       = 000000C0
DAP$K_FIX       = 00000001
DAP$K_SEQ       = 00000000
DAP$L_ALQ1      = 0000004C
DAP$L_ATTMENU   = 00000040
DAP$L_CMWA      = 00000030
DAP$L_CRC_RSLT  = 00000020
DAP$L_DCODE_STS = 00000018
DAP$L_DEV       = 00000068
DAP$L_EBK       = 00000078
DAP$L_FOP1      = 00000064
DAP$L_HBK       = 00000074
DAP$L_MRN       = 00000058
DAP$L_MSG_MASK  = 0000001C
DAP$L_SBN       = 0000007C
DAP$L_SSPWA     = 00000080
DAP$L_TEMP      = 00000090
DAP$M_CMPFMT    = 00000008
DAP$M_DMO       = 00002000
DAP$M_EMBEDDED  = 00000010
DAP$M_IMAGE     = 00000002
DAP$M_LSA       = 00000040
DAP$M_MACY11    = 00000080
DAP$M_TMP1$     = 00000040
DAP$M_TMP2$     = FC000000
DAP$M_TMP3$     = 00020000
DAP$M_TMP4$     = 01000000
DAP$M_TMP5$     = F0000000
DAP$M_ZERO      = 00000080
DAP$Q_DCODE_FLG = 00000000
DAP$Q_MSG_BUF1  = 00000008
DAP$Q_MSG_BUF2  = 00000010
DAP$Q_RUNSYS    = 0000005C
DAP$V_DEV       = 0000000E
DAP$V_DEVALL    = 0000000C
DAP$V_DEVAVL    = 00000010
DAP$V_DEVCCL    = 00000001
DAP$V_DEVDIR    = 00000003
DAP$V_DEVDMT    = 0000000B

```

```

DAP$V_DEVELG    = 00000011
DAP$V_DEVFOD    = 00000007
DAP$V_DEVFOR    = 00000017
DAP$V_DEVGEN    = 00000019
DAP$V_DEVIDV    = 0000000D
DAP$V_DEVMBX    = 00000012
DAP$V_DEVMNT    = 0000000A
DAP$V_DEVNET    = 00000018
DAP$V_DEVODV    = 0000000E
DAP$V_DEVRCK    = 00000015
DAP$V_DEVREC    = 00000000
DAP$V_DEVRND    = 00000014
DAP$V_DEVRTM    = 00000013
DAP$V_DEVSDI    = 00000004
DAP$V_DEVSHR    = 00000008
DAP$V_DEVSPL    = 00000009
DAP$V_DEVSQD    = 00000005
DAP$V_DEVSWL    = 0000000F
DAP$V_DEVTRM    = 00000002
DAP$V_DEVWCK    = 00000016
DAP$V_GEQ_V56   = 00000024
DAP$V_VAXELAN   = 00000035
DAP$V_VAXVMS    = 00000034
DAP$W_BLS       = 00000048
DAP$W_DEQ1      = 00000054
DAP$W_FFB       = 00000072
DAP$W_LRL       = 00000070
DAP$W_MRS       = 0000004A
DAP$W_PARTNER   = 00000006
DAP$W_VERSION   = 00000004
DEV$V_ALL       = 00000017
DEV$V_AVL       = 00000012
DEV$V_CCL       = 00000001
DEV$V_DIR       = 00000003
DEV$V_DMT       = 00000015
DEV$V_ELG       = 00000016
DEV$V_FOD       = 0000000E
DEV$V_FOR       = 00000018
DEV$V_GEN       = 00000011
DEV$V_IDV       = 0000001A
DEV$V_MBX       = 00000014
DEV$V_MNT       = 00000013
DEV$V_NET       = 0000000D
DEV$V_ODV       = 0000001B
DEV$V_RCK       = 0000001E
DEV$V_REC       = 00000000
DEV$V_RND       = 0000001C
DEV$V_RTM       = 0000001D
DEV$V_SDI       = 00000004
DEV$V_SHR       = 00000010
DEV$V_SPL       = 00000006
DEV$V_SQD       = 00000005
DEV$V_SWL       = 00000019
DEV$V_TRM       = 00000002
DEV$V_WCK       = 0000001F
ERRQU0          = 000000AF
FAP$B_ACMODES   = 0000004A

```

R 01

NT
VO

NTOACCESS
Symbol table

NETWORK ACCESS/DEACCESS

L 6

15-SEP-1984 23:47:02 VAX/VMS Macro V04-00
5-SEP-1984 16:20:08 [RMS.SRC]NTOACCESS.MAR;1

Page 20
(9)

FABSL_DEV	= 00000040		
FABSL_FOP	= 00000004		
FABSL_NAM	= 00000028		
FABSL_SDC	= 00000044		
FABSS_CHAN_MODE	= 00000002		
FABSV_CHAN_MODE	= 00000002		
FABSV_UFO	= 00000011		
FWASB_SUBNODCNT	= 0000002F		
FWASQ_NODE1	= 000001B4		
FWASQ_QUOTED	= 00000170		
FWASV_NETSTR	= 00000032		
FWASV_OBJTYPE	= 00000031		
FWASV_QUOTED	= 0000001A		
FWASV_WILDCARD	= 00000018		
IFBSB_MODE	= 0000000A		
IFBSL_AS_DEV	= 0000008C		
IFBSL_DEVBUFSIZ	= 00000048		
IFBSL_IOS	= 0000000C		
IFBSL_NWA_PTR	= 0000003C		
IFBSL_PRIM_DEV	= 00000000		
IFBSV_ACCESSED	= 00000025		
IFBSV_DAP	= 0000003E		
IFBSV_NSP	= 0000003F		
IFBSW_CHNL	= 00000020		
IOSM_ACCESS	= 00000040		
IOSM_SYNCH	= 00000200		
IOS_ACCESS	= 00000032		
IOS_DEACCESS	= 00000034		
NAMSL_FNB	= 00000034		
NAMSV_WILDCARD	= 00000008		
NTSACCESS	00000201	RG	01
NTSASSIGN	00000000	RG	01
NTSDEACCESS	00000200	RG	01
NTSMAP_DEV_CHAR	000000E6	RG	01
NTSMOD_DEV_CHAR	000000B9	RG	01
NTSRET_DEV_CHAR	000001D4	RG	01
NWASB_ALLXABCNT	0000011C		
NWASB_DAP_RAC	000000C9		
NWASB_FILESYS	000000C5		
NWASB_KEYXABCNT	0000011D		
NWASB_NETSTRSIZ	0000016F		
NWASB_NODBUFSIZ	00000168		
NWASB_ORG	000000C6		
NWASB_OSTYPE	000000C4		
NWASB_RFM	000000C7		
NWASB_RMS_RAC	000000C8		
NWASC_BLN	00000800		
NWASK_BLN	00000800		
NWASK_SEQ	= 00000000		
NWASL_ALLXABADR	00000100		
NWASL_DATXABADR	00000104		
NWASL_DEV	000000C0		
NWASL_FHCXABADR	00000108		
NWASL_KEYXABADR	0000010C		
NWASL_MSG_MASK	000000D4		
NWASL_PROXABADR	00000110		
NWASL_RDXABADR	00000114		

NWASL_SAVE_FLGS	00000128		
NWASL_SUMXABADR	00000118		
NWASL_THREAD	000000FC		
NWASL_XLTATTR	00000238		
NWASL_XLTBUF1	0000022C		
NWASL_XLTCNT	00000228		
NWASL_XLTMAXINDX	00000234		
NWASL_XLTSIZ	00000230		
NWASQ_ACS	00000244		
NWASQ_BIGBUF	00000170		
NWASQ_BLD	000000F0		
NWASQ_FLG	00000000		
NWASQ_INODE	0000025C		
NWASQ_IOSB	000000D8		
NWASQ_LNODE	00000160		
NWASQ_LOGNAME	0000023C		
NWASQ_NCB	00000264		
NWASQ_RCV	000000E0		
NWASQ_SAVE_DESC	00000120		
NWASQ_XLTBUF1	0000024C		
NWASQ_XLTBUF2	00000254		
NWASQ_XMT	000000E8		
NWAST_ACSBUF	0000026C		
NWAST_AUXBUF	000005E0		
NWAST_DAP	00000000		
NWAST_INODEBUF	000004AC		
NWAST_ITM_ATTR	00000200		
NWAST_ITM_END	00000224		
NWAST_ITM_LST	00000200		
NWAST_ITM_MAXINDX	00000218		
NWAST_ITM_STRING	0000020C		
NWAST_NCBBUF	0000052C		
NWAST_NODEBUF	00000169		
NWAST_RCVBUF	000001A0		
NWAST_SCAN	00000100		
NWAST_TEMP	00000120		
NWAST_XLTBUF1	000002AC		
NWAST_XLTBUF2	000003AC		
NWAST_XMTBUF	000003C0		
NWASV_DEVCHAR	= 00000011		
NWASV_DEVMBX	= 00000013		
NWASV_DEVTRM	= 00000010		
NWASW_BUILD	000000D2		
NWASW_DAPBUFSIZ	000000CA		
NWASW_DIR_OFF	000000CC		
NWASW_DISPLAY	000000D0		
NWASW_FIL_OFF	000000CE		
NWASW_JNLXABJOP	0000011E		
PARSE_QUOTED_STRING	00000052	R	01
PIOSA_TRACE	*****	X	01
PSLSC_EXEC	= 00000001		
RMSCHNAM	*****	X	01
RMSSETEFN	*****	X	01
RMSSTALL	*****	X	01
RMSSTALLAST	*****	X	01
RMS QUO	= 00018634		
SYSSASSIGN	*****	GX	01

NT
VO

NTOACCESS
Symbol table

NETWORK ACCESS/DEACCESS

M 6

15-SEP-1984 23:47:02
5-SEP-1984 16:20:08

VAX/VMS Macro V04-00
[RMS.SRC]NTOACCESS.MAR;1

Page 21
(9)

SYSSQIO
TPTSL_NTACC_DAP
TPTSL_NTACC_NSP
TPTSL_NTDEACCES

***** GX 01
***** X 01
***** X 01
***** X 01

+-----+
! Psect synopsis !
+-----+

PSECT name

Allocation

PSECT No.

Attributes

. ABS
NFSNETWORK
\$ABSS

00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
000002CE (718.)	01 (1.)	PIC	USR	CON	REL	GBL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE
00000800 (2048.)	02 (2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	47	00:00:00.15	00:00:01.17
Command processing	154	00:00:00.82	00:00:05.87
Pass 1	466	00:00:19.19	00:00:47.35
Symbol table sort	0	00:00:02.71	00:00:03.95
Pass 2	142	00:00:03.55	00:00:08.10
Symbol table output	28	00:00:00.18	00:00:00.50
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	841	00:00:26.62	00:01:06.96

The working set limit was 1800 pages.
105919 bytes (207 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1925 non-local and 57 local symbols.
768 source lines were read in Pass 1, producing 15 object records in Pass 2.
35 pages of virtual memory were used to define 34 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name

Macros defined

-\$255\$DUA28:[RMS.OBJ]RMS.MLB;1
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2
TOTALS (all libraries)

17
13
30

2224 GETS were required to define 30 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:NTOACCESS/OBJ=OBJ\$:NTOACCESS MSRC\$:NTOACCESS/UPDATE=(ENH\$:NTOACCESS)+LIB\$:RMS/LIB

0315 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NT0ACCESS
LIS

NT0CLOSE
LIS

NT0BLDXAB
LIS

NT0CONN
LIS

NT0CREATE
LIS

NT0DAP10
LIS

NT0DAPCRC
LIS

NT0ACFIL
LIS

NT0BLK10
LIS